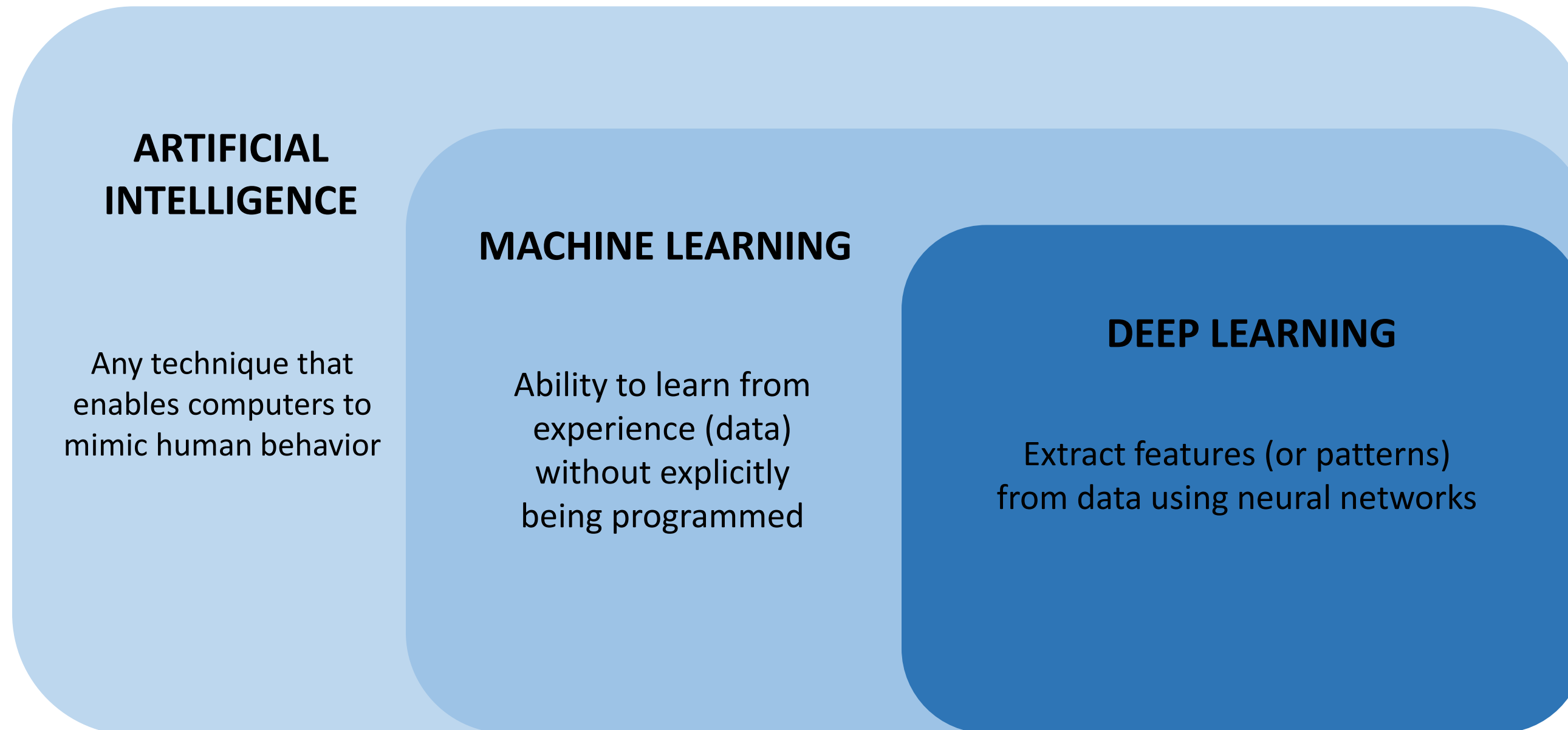




# Introduction to Deep Learning

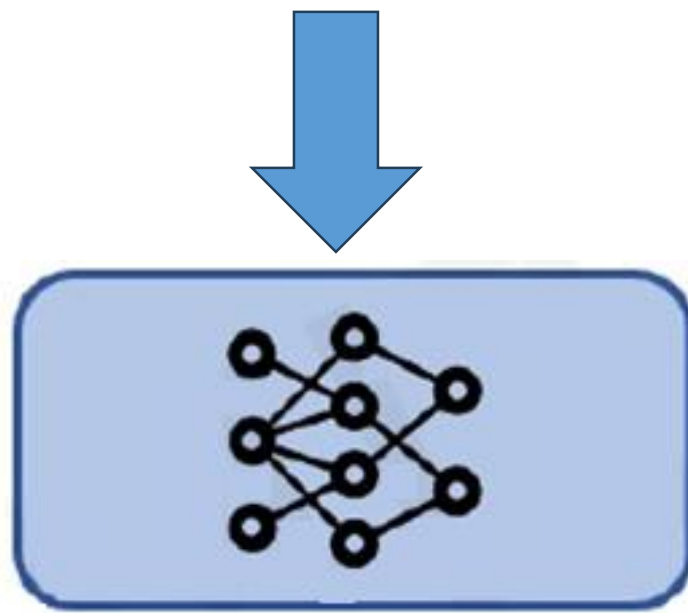
# What is Deep Learning?



- Deep Learning is accelerating at a rate faster than we have never seen before !!

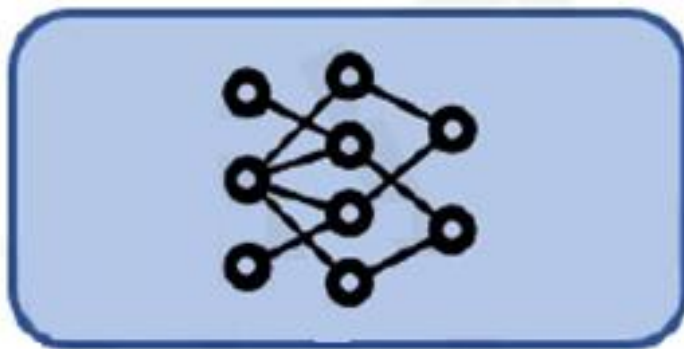
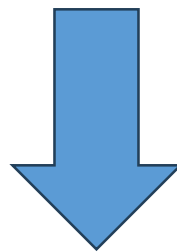
# Generating Images from Natural Language

“A photo of an astronaut riding a horse”



# Generating Language from Natural Language

“Write a code in TensorFlow to train a neural network”



Certainly! Here is an example of how you can use TensorFlow to train a neural network for binary classification:

```
import tensorflow as tf

# Load the data
(x_train, y_train), (x_test, y_test) = load_data()

# Define the model
model = tf.keras.Sequential([
    tf.keras.layers.Input(shape=x_train.shape[1:]),
    tf.keras.layers.Dense(16, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=
['accuracy'])

# Train the model
model.fit(x_train, y_train, epochs=10, batch_size=32, validation_data=
(x_test, y_test))
```

This code assumes that you have a function `load_data()` that returns the training and test datasets as tuples of NumPy arrays. The input shape of the model is determined by the shape of the training data, and the model has two dense layers with 16 and 1 units, respectively, with ReLU and sigmoid activations. The model is then compiled with an Adam optimizer and a binary cross-entropy loss function, and is trained using the `fit()` method.

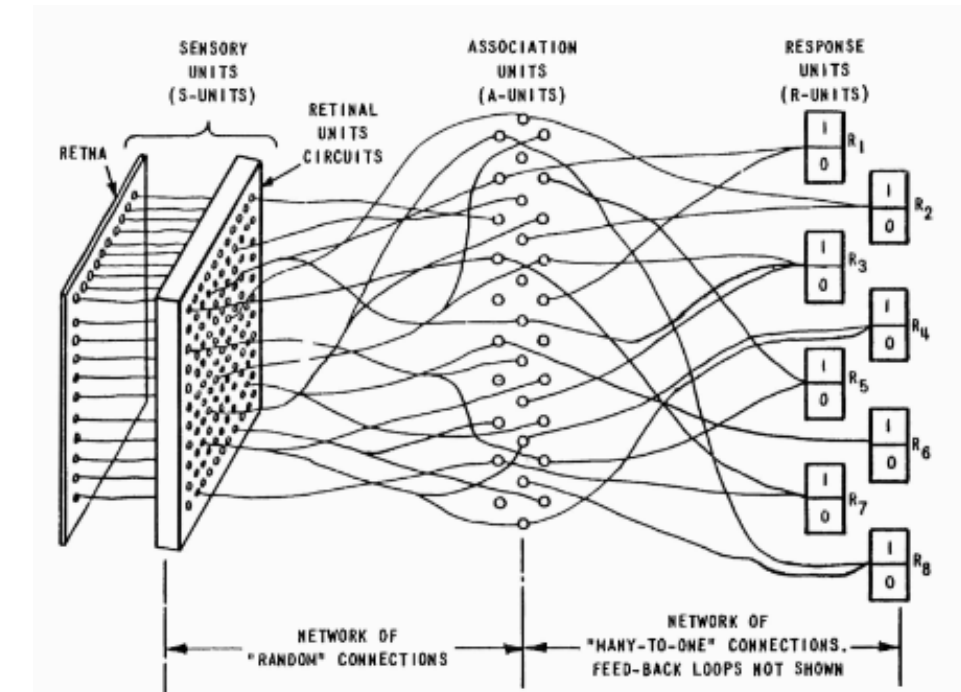
I hope this helps! Let me know if you have any questions.

# Artificial Intelligence in the context of Human History



**Alan Turing, 1951:** "It seems probable that once the machine thinking method had started, it would not take long to outstrip our feeble powers. They would be able to converse with each other to sharpen their wits. At some stage therefore, we should have to expect the machines to take control."

# Artificial Intelligence in the context of Human History



**Frank Rosenblatt (1928–1971):** was an American psychologist and computer scientist who made significant contributions to the field of artificial intelligence and neural networks. He is best known for developing the **perceptron**, a type of artificial neural network model, and for his work on pattern recognition and machine learning.

# Artificial Intelligence in the context of Human History



Kasparov vs. Deep Blue, 1997

# Artificial Intelligence in the context of Human History



Lee Sedol vs. AlphaGo, 2016

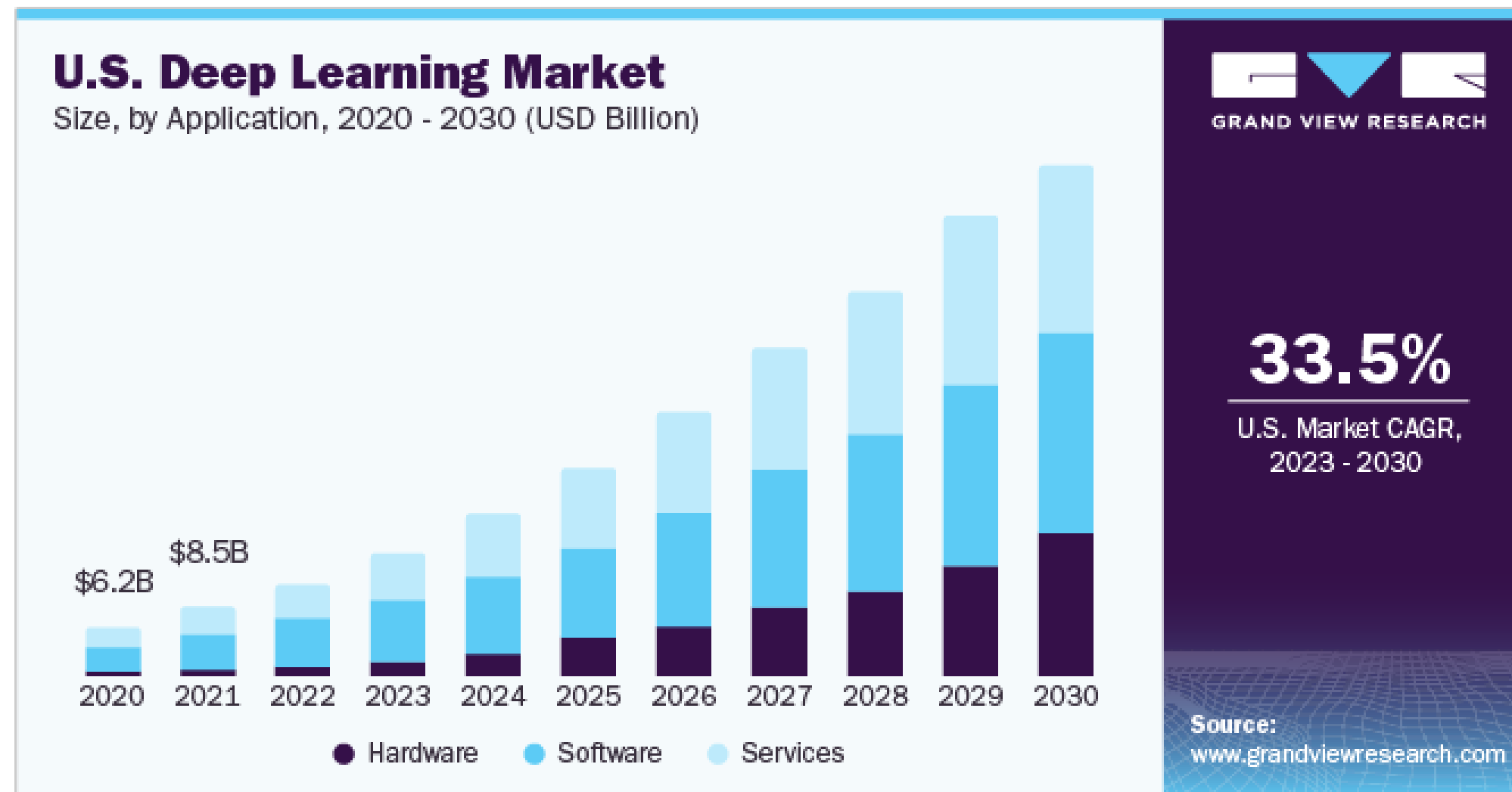
- AlphaGo is a computer program developed by DeepMind, a subsidiary of Alphabet Inc. (Google's parent company)<sup>8</sup>

# History of Deep Learning Ideas and Milestones

- 1943: Neural networks
- 1957-62: Perceptron
- 1970-86: Backpropagation, RNN
- 1979-98: CNN, MNIST, LSTM, Bidirectional RNN
- 2006: “Deep Learning”, DBN
- 2009: ImageNet + AlexNet
- 2014: GANs
- 2016-17: AlphaGo, AlphaZero
- 2017-19: Transformers
- 2020: GPT-3
- 2022: ChatGPT (Chat Generative Pre-trained Transformer)



# Deep Learning Market



# Why Deep Learning now?

Neural Networks date back decades. So why now ?

## 1. Big Data

- Larger Datasets
- Easier Collection and Storage

## 2. Hardware

- Graphics Processing Units (GPUs)
- Massively Parallelizable

## 3. Software

- Improved techniques
- New Models
- Toolboxes

IMAGENET



WIKIPEDIA  
The Free Encyclopedia

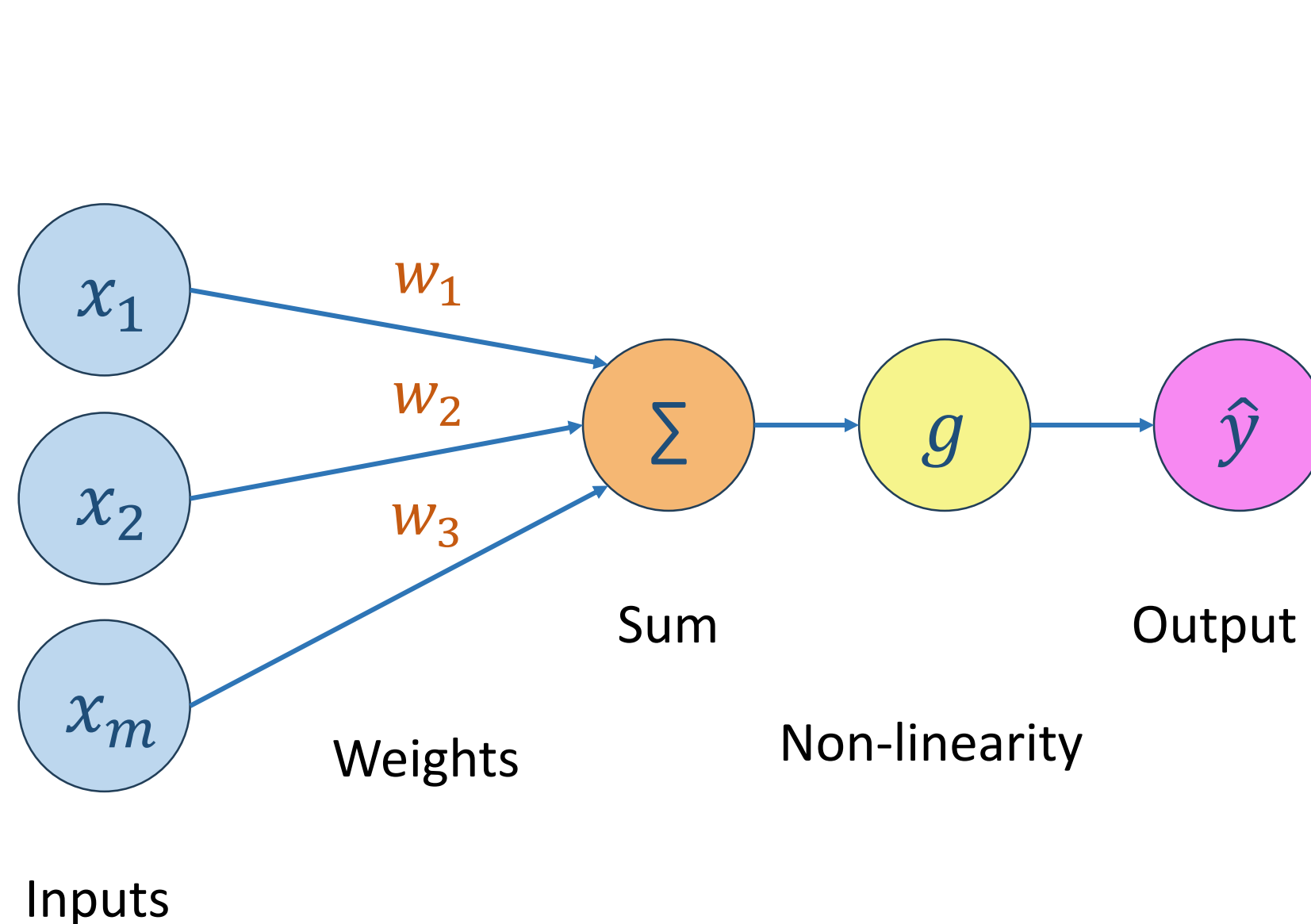


PyTorch



TensorFlow

# The Perceptron: Forward Propagation



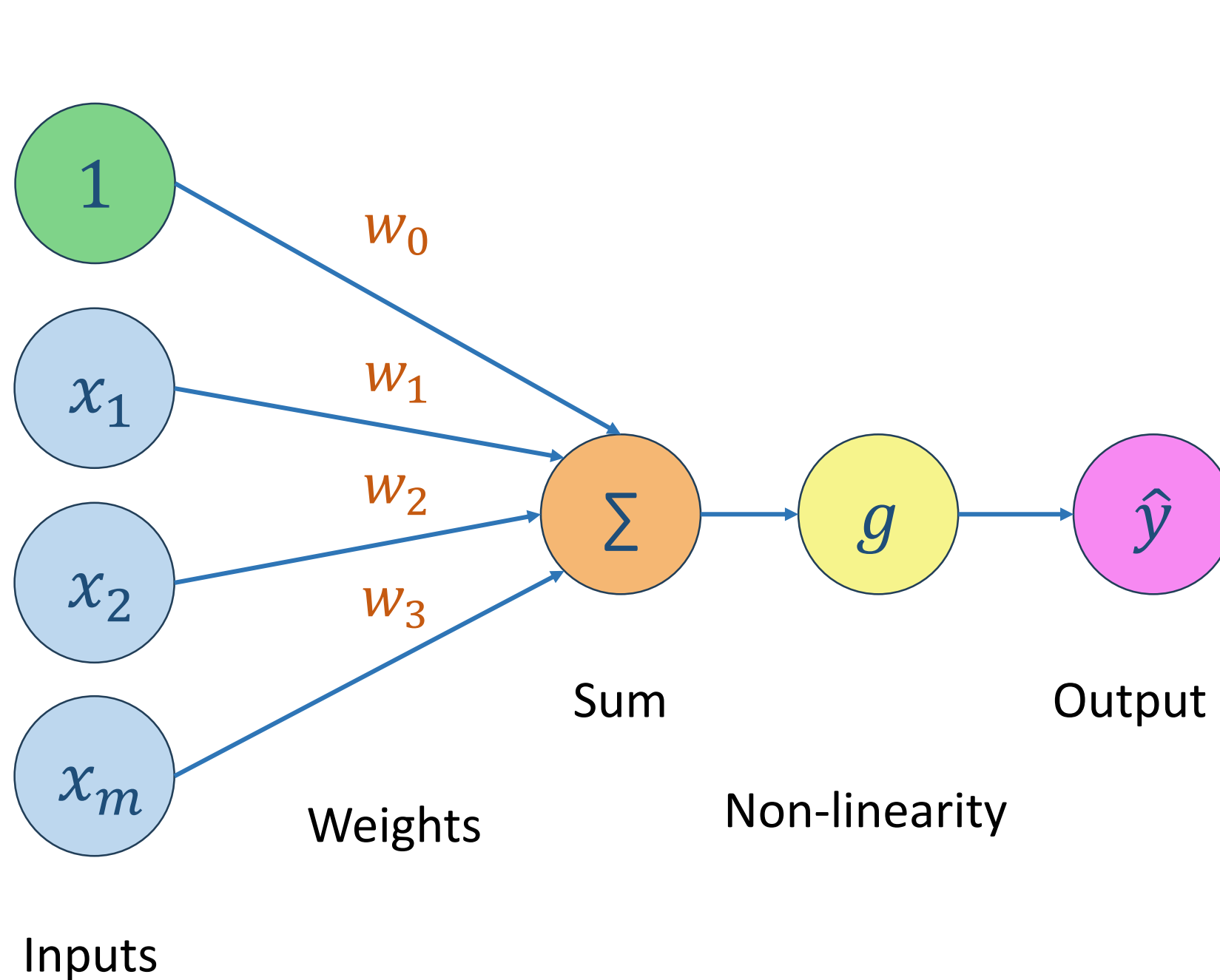
Linear combination of inputs

Output

$$\hat{y} = g \left( \sum_{i=1}^m x_i w_i \right)$$

Non-linear activation function

# The Perceptron: Forward Propagation



Linear combination of inputs

Output

$$\hat{y} = g \left( w_0 + \sum_{i=1}^m x_i w_i \right)$$

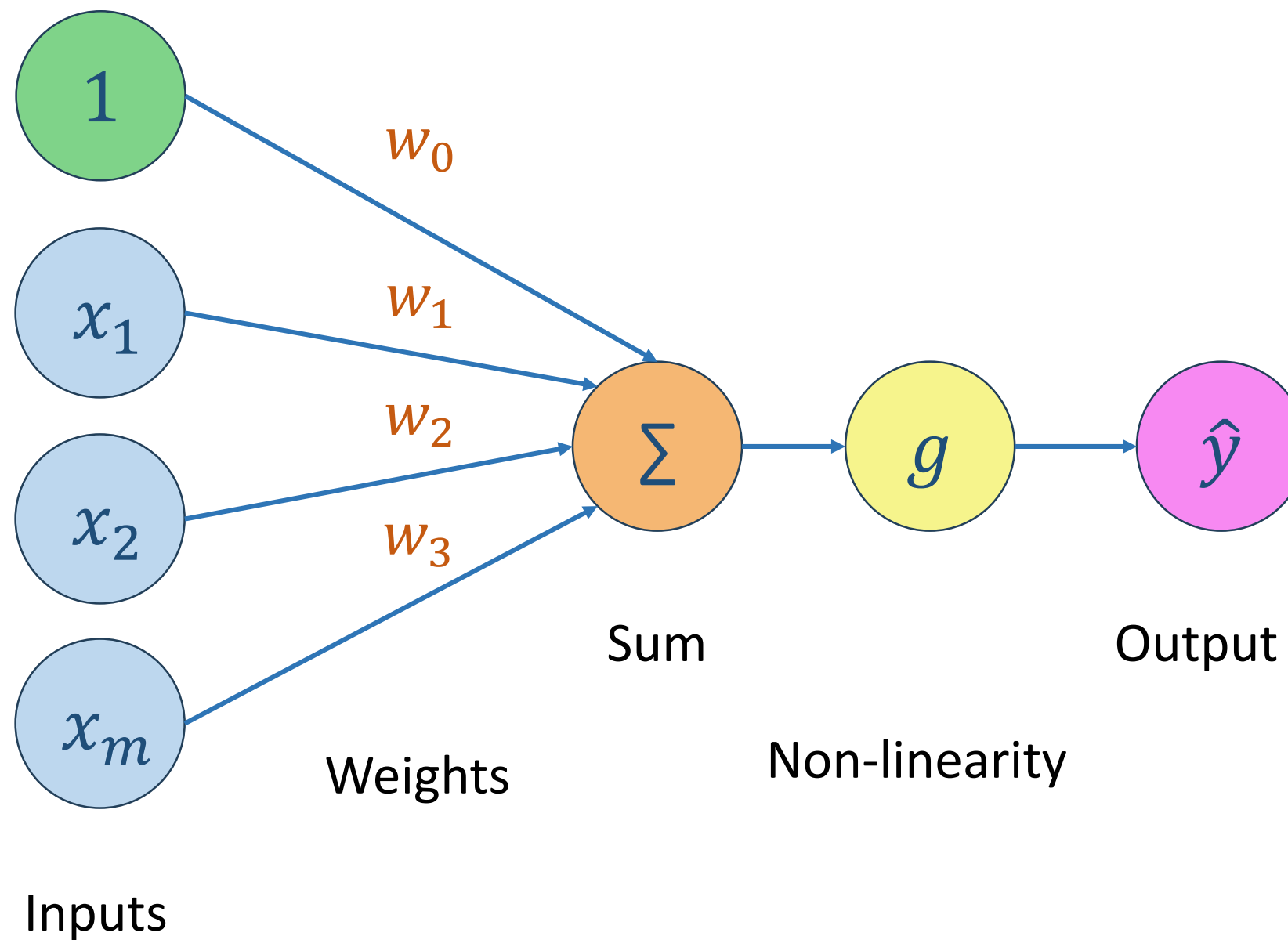
Non-linear activation function

Bias

$$\hat{y} = g (w_0 + X^T W)$$

where  $X = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$  and  $W = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}$

# The Perceptron: Forward Propagation

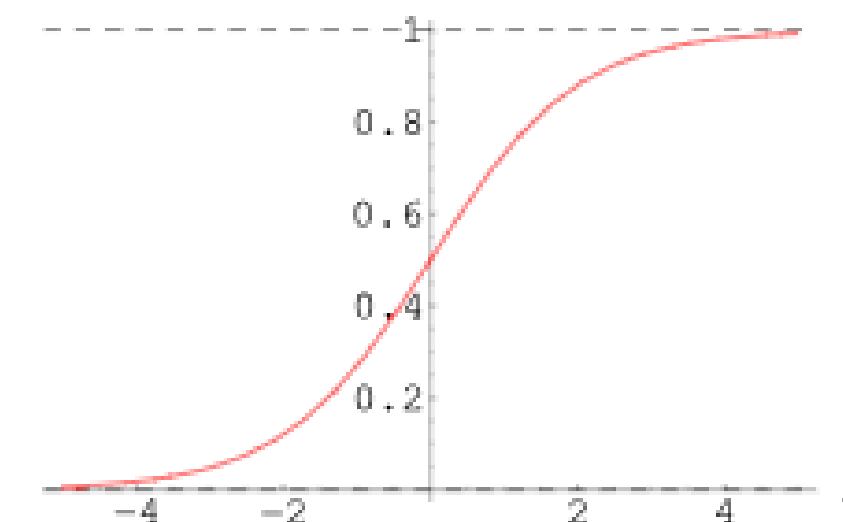


Activation Function:

$$\hat{y} = g \left( w_0 + \sum_{i=1}^m x_i w_i \right)$$

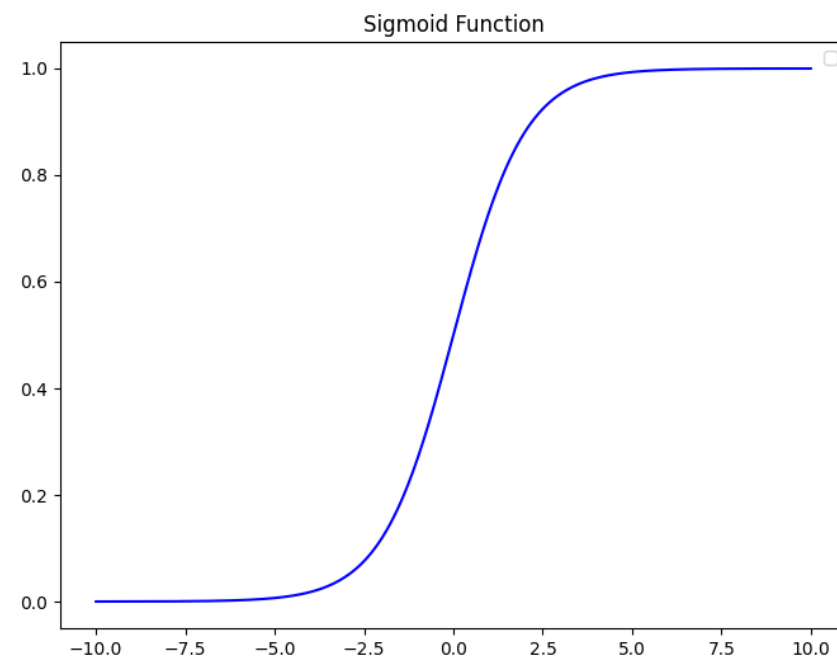
Example: Sigmoid function

$$g(z) = \sigma(z) = \frac{1}{1 + e^{-z}}$$



# Common Activation Functions

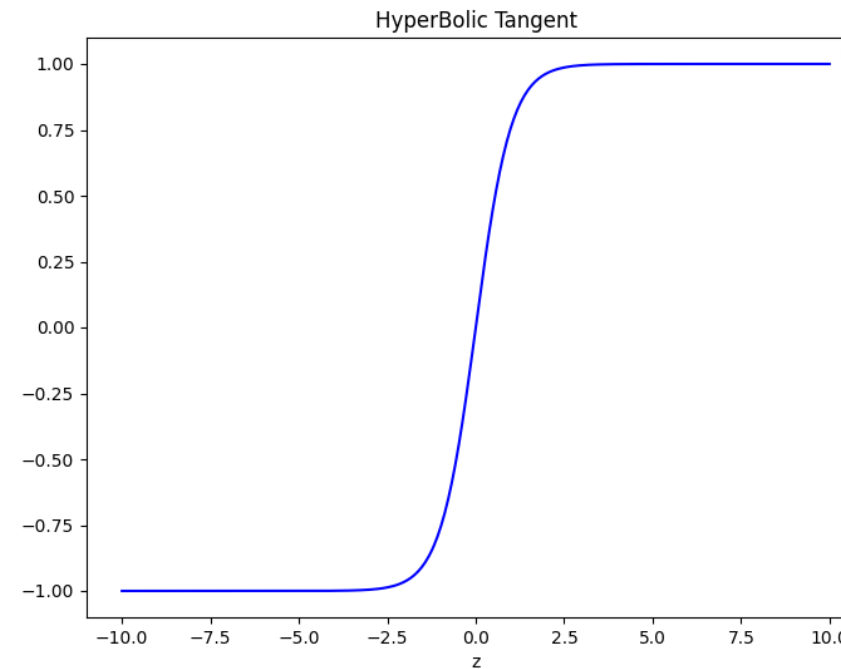
### Sigmoid Function



$$g(z) = \frac{1}{1 + e^{-z}}$$

 `tf.math.sigmoid(z)`

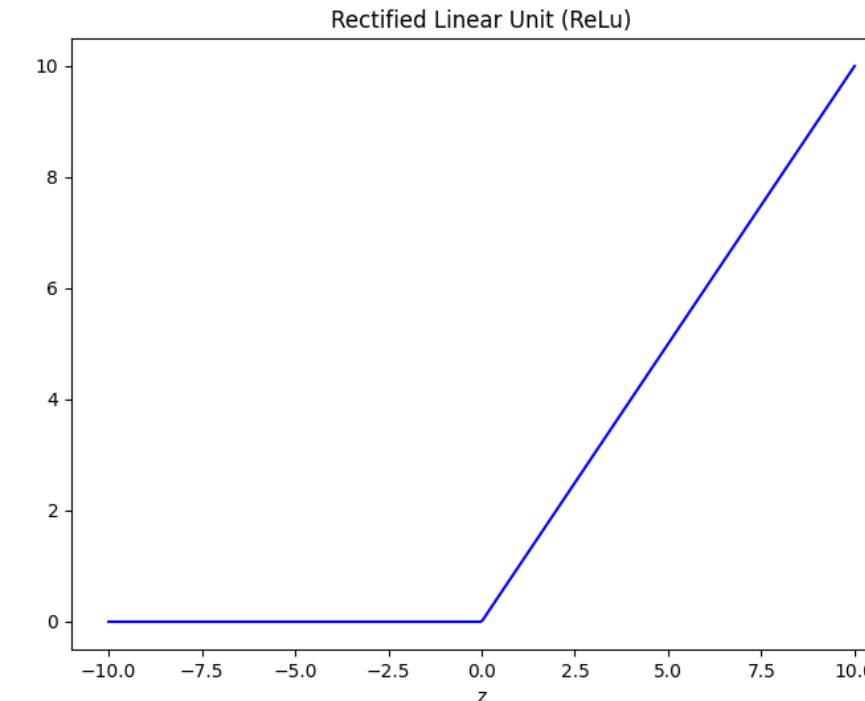
### HyperBolic Tangent



$$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$$

 `tf.math.tanh(z)`

### Rectified Linear Unit (ReLU)

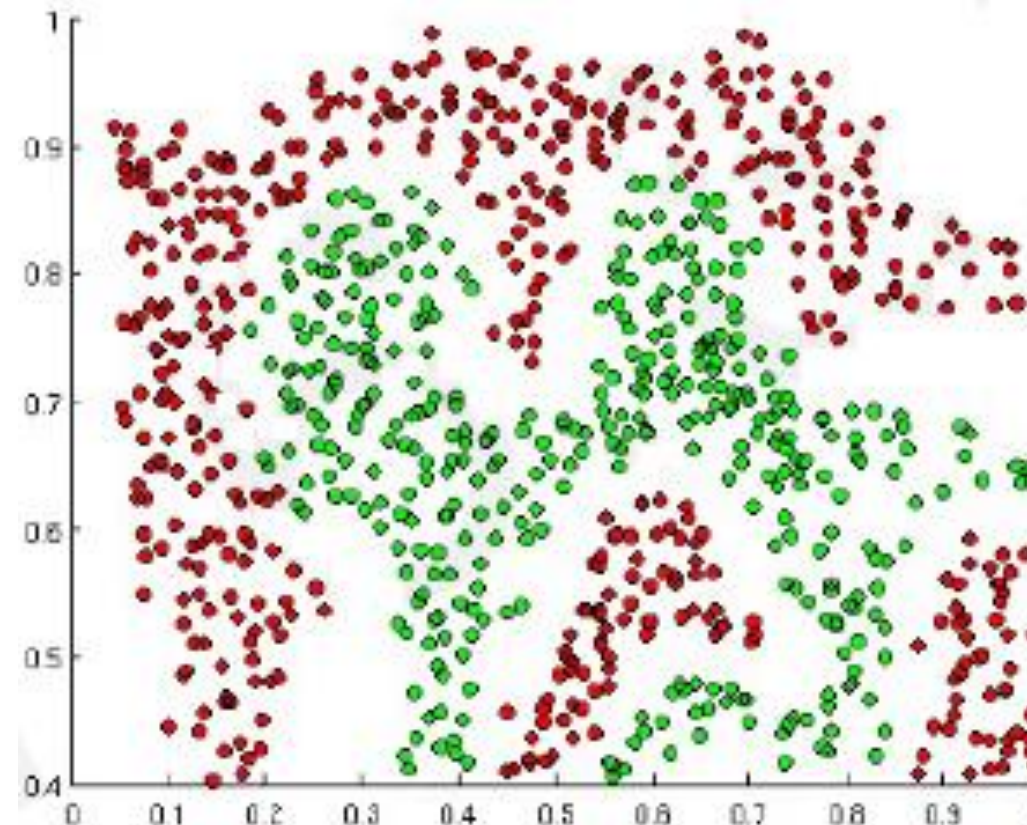


$$g(z) = \max(0, z)$$

 `tf.nn.relu(z)`

# Importance of Activation Functions

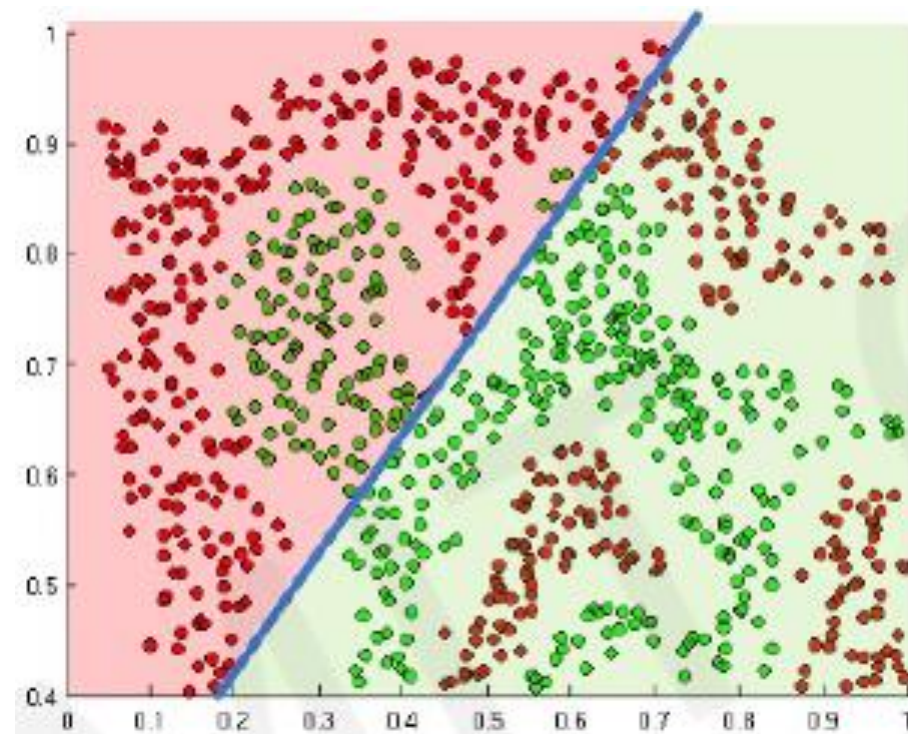
- The purpose of activation functions is to introduce non-linearities in the network



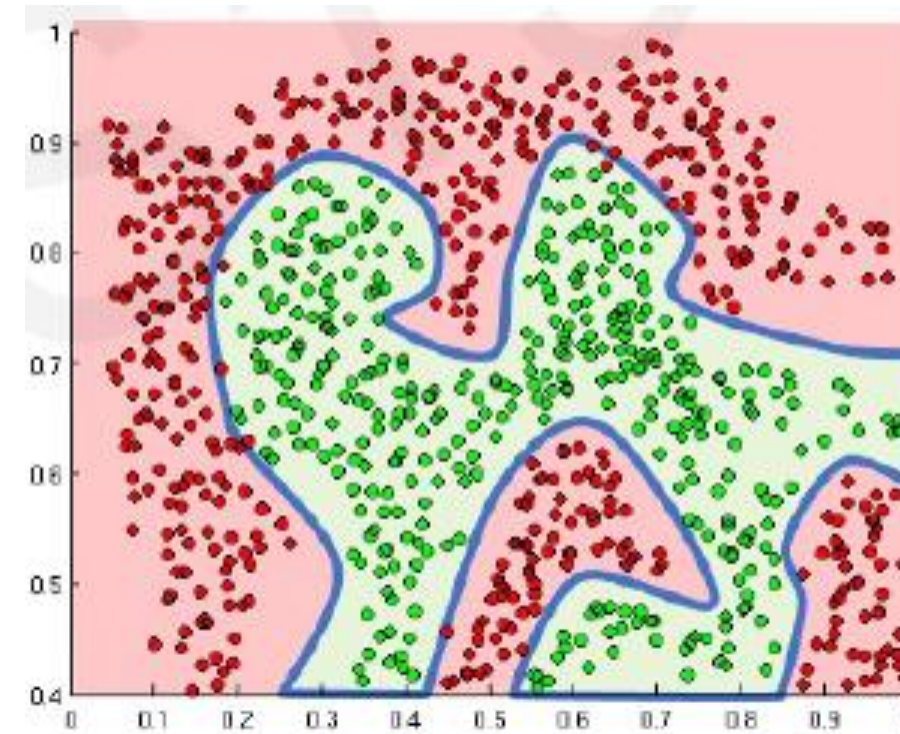
What if we wanted to build a neural network to distinguish green vs red points?

# Importance of Activation Functions

- The purpose of activation functions is to introduce non-linearities in the network



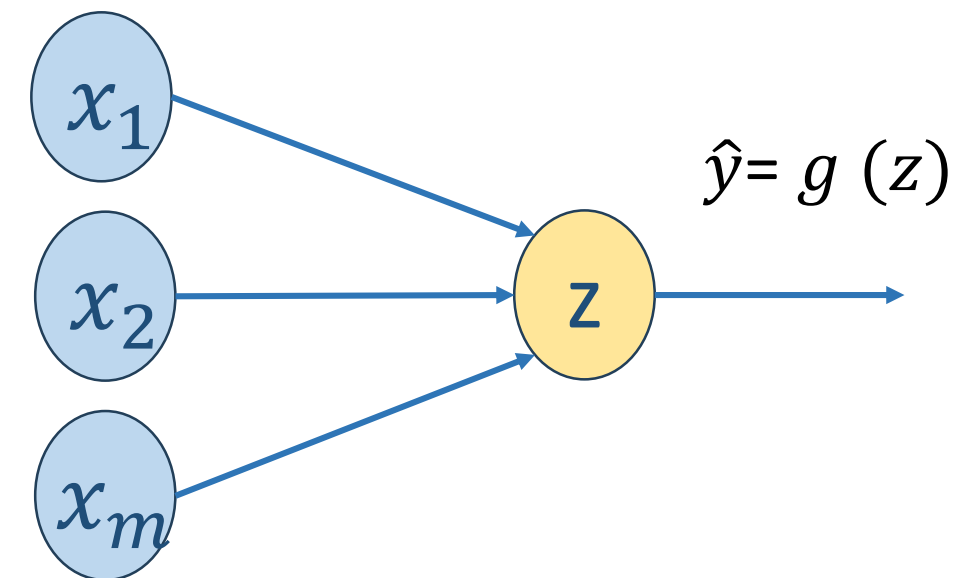
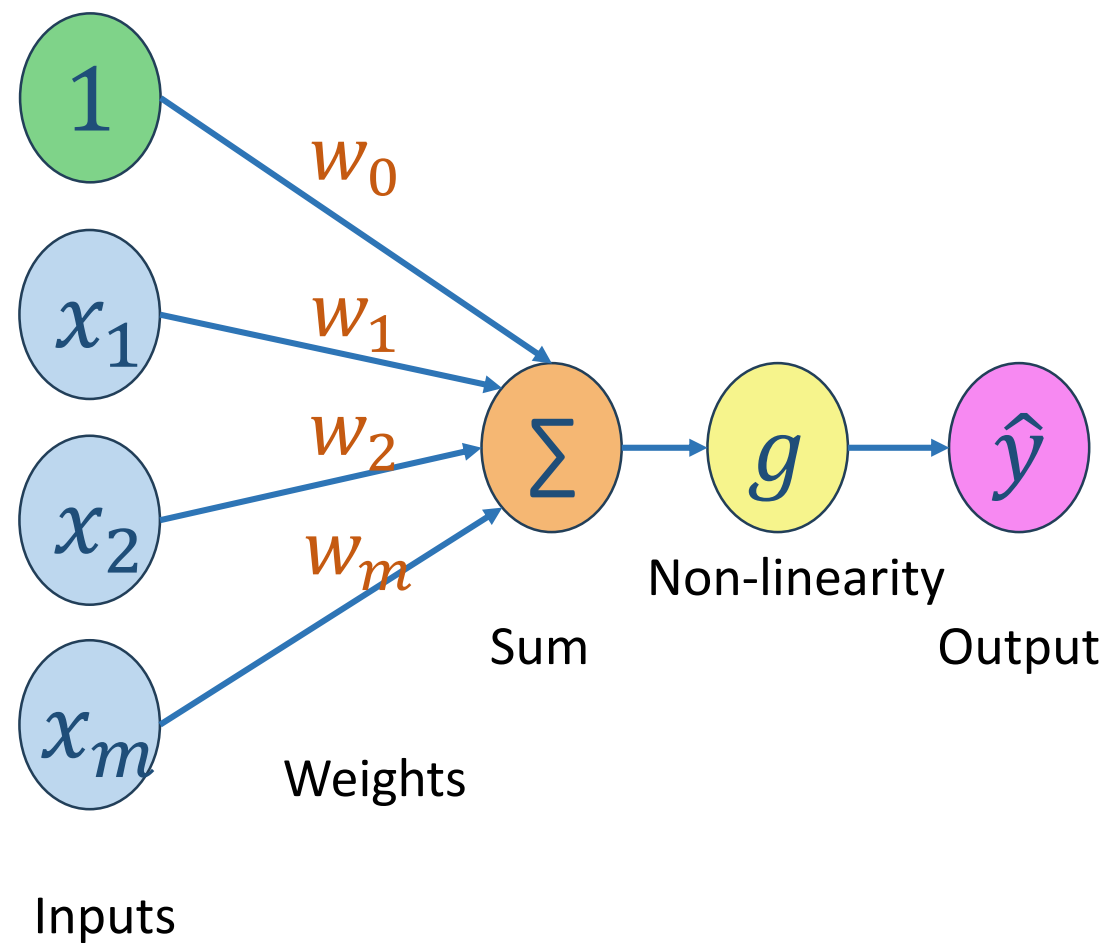
Linear activation functions  
produce linear decisions no  
matter the network size



Non-linearities allow us to  
approximate arbitrary complex  
functions

# The Perceptron: Simplified

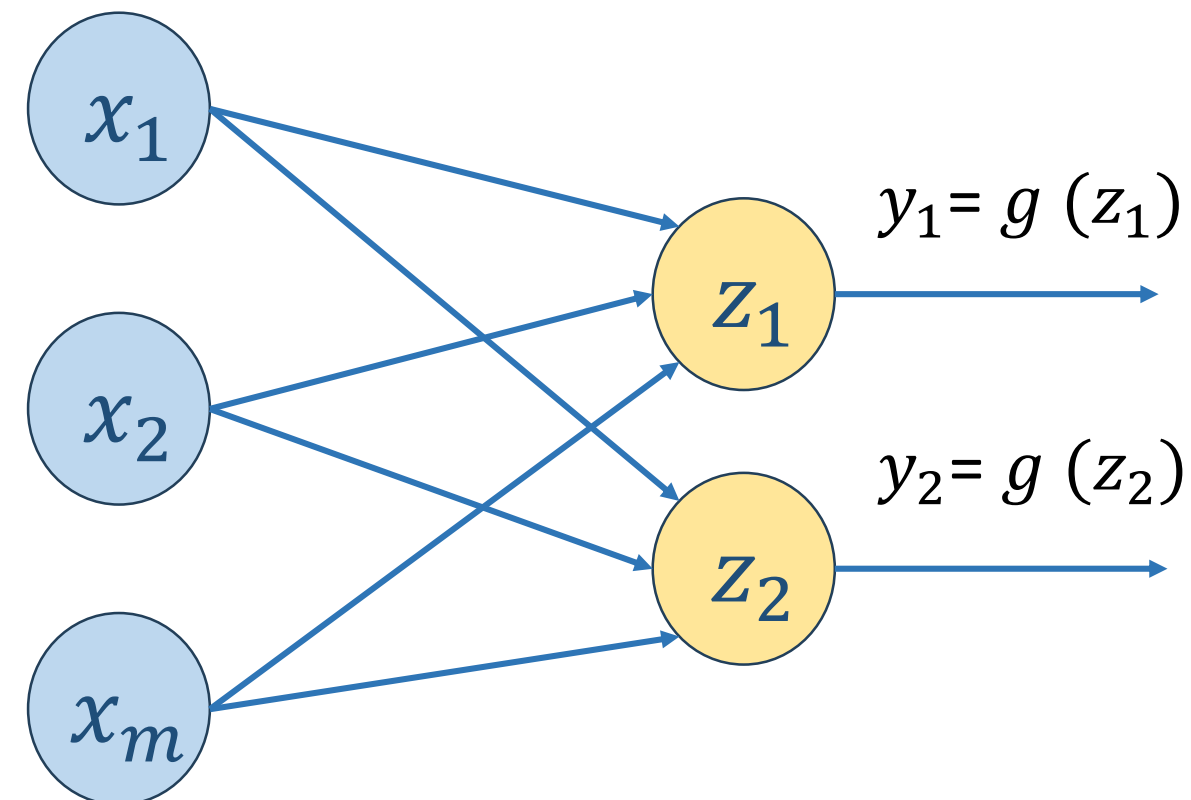
$$\hat{y} = g(w_0 + X^T W)$$



$$z = w_0 + \sum_{j=1}^m x_j w_j$$

# Multi Output Perceptron

Because all inputs are densely connected to all outputs, these layers are called Dense layers



$$z_i = w_{0,i} + \sum_{j=1}^m x_j w_{j,i}$$

# Dense Layer from scratch

```
import tensorflow as tf

class MyDenseLayer (tf.keras.layers.Layer):
    def __init__(self, input_dim, output_dim):
        super(MyDenseLayer, self).__init__()

        # Initialize weights and bias
        self.W=self.add_weight([input_dim, output_dim])
        self.b=self.add_weight([1,output_dim])

    def call(self, inputs):

        # Forward propagate the inputs
        z=tf.matmul(inputs,self.W)+self.b

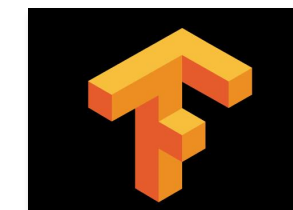
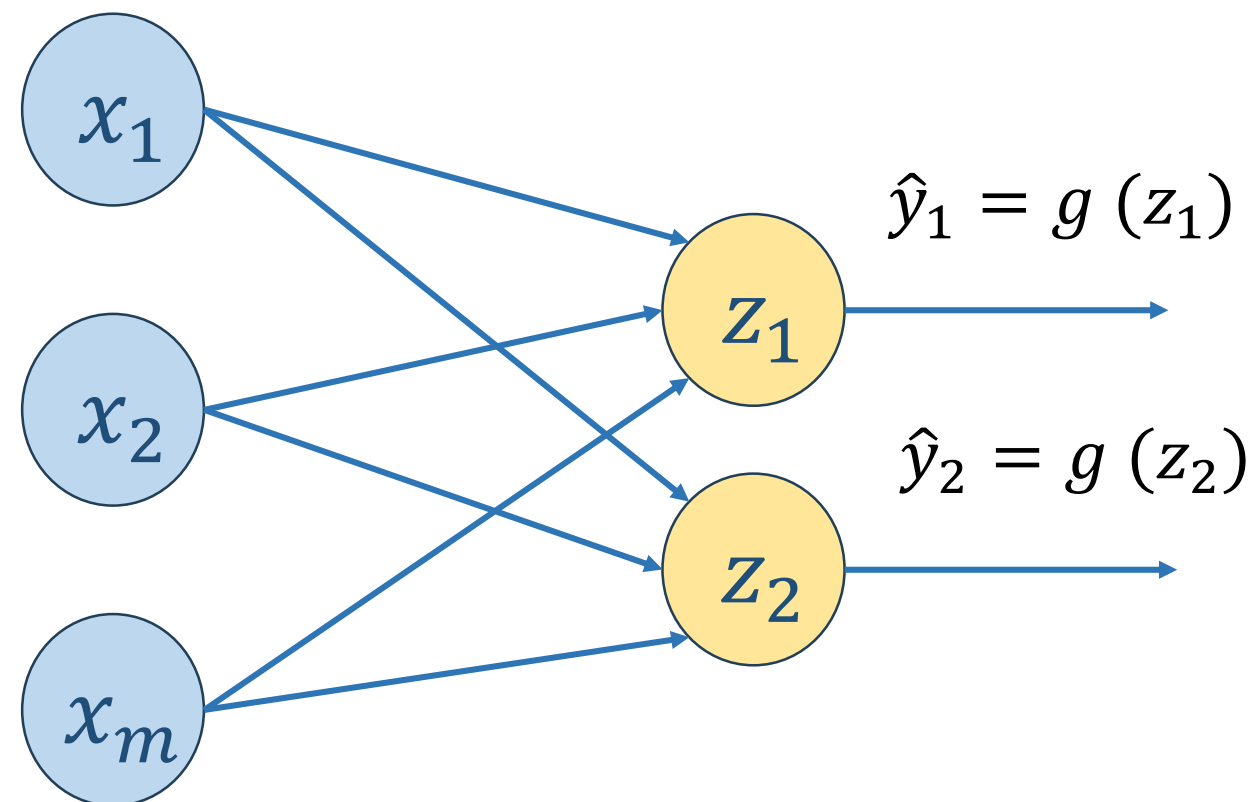
        # Feed through a non-linear activation function
        output = tf.math.sigmoid(z)

        return output
```



# Multi Output Perceptron

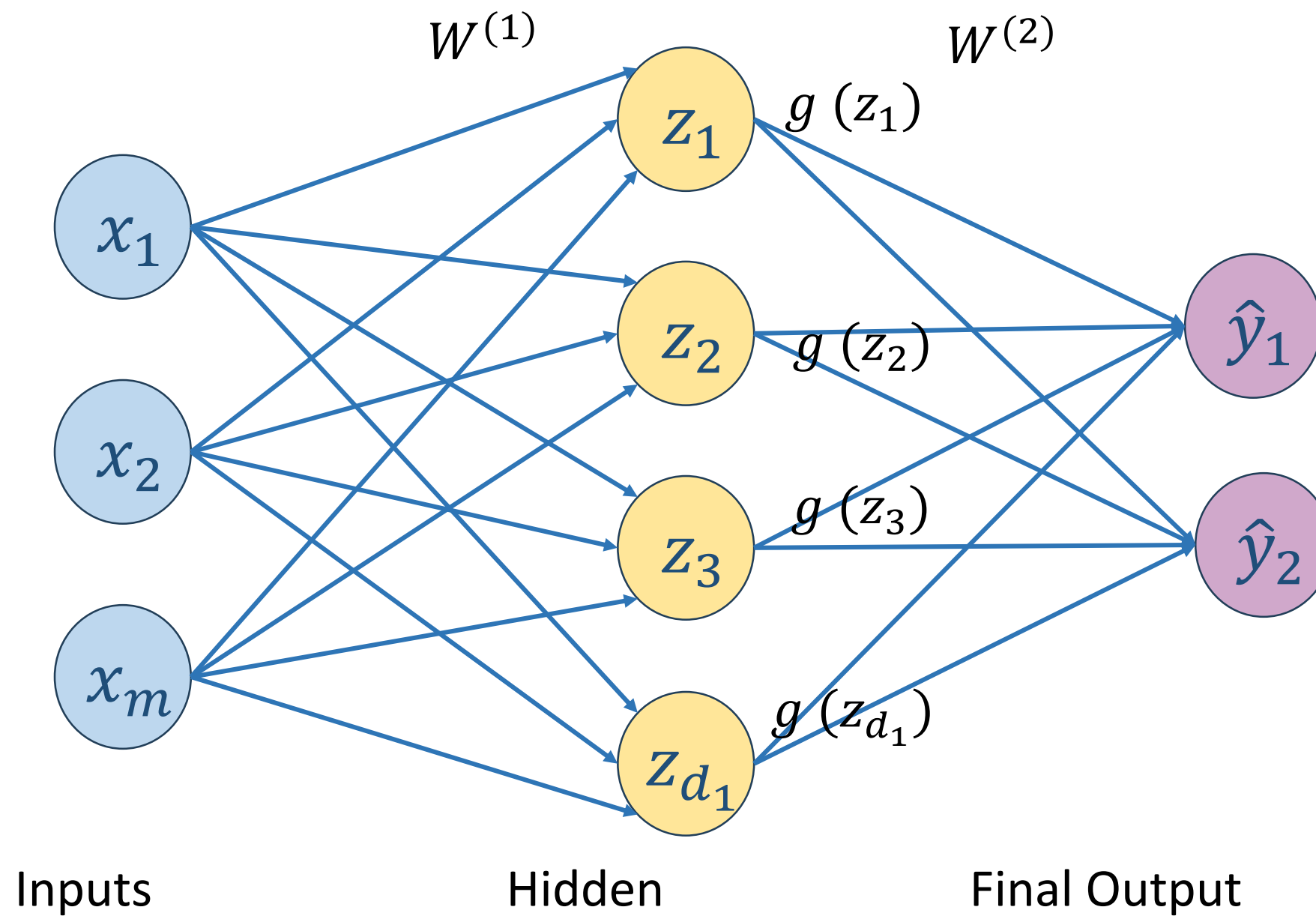
Because all inputs are densely connected to all outputs, these layers are called Dense layers



```
import tensorflow as tf  
  
layer = tf.keras.layers.Dense(units = 2)
```

$$z_i = w_{0,i} + \sum_{j=1}^m x_j w_{j,i}$$

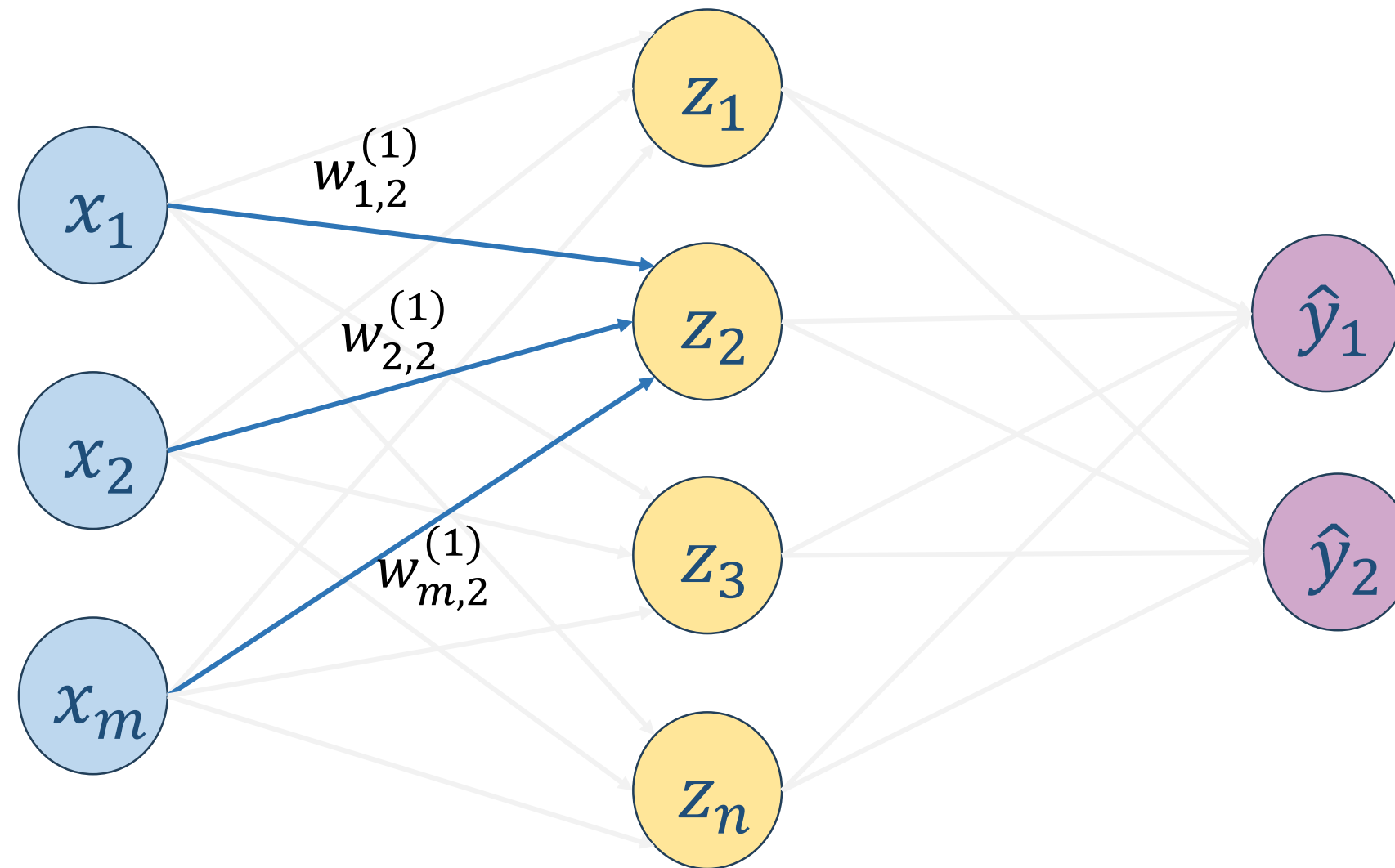
# Hidden Layers



$$z_i = w_{0,i}^{(1)} + \sum_{j=1}^m x_j w_{j,i}^{(1)}$$

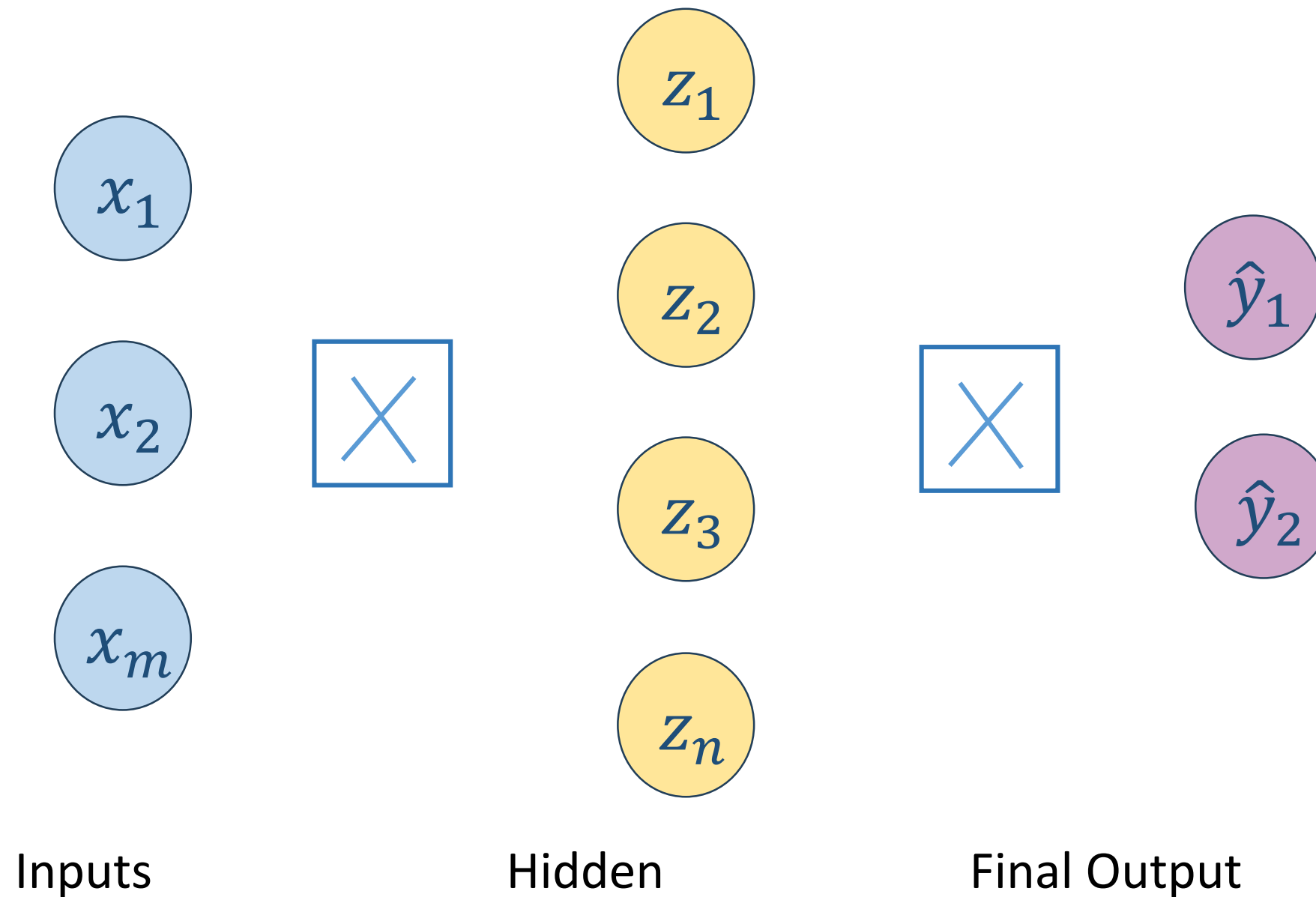
$$\hat{y}_i = g \left( w_{0,i}^{(2)} + \sum_{j=1}^{d_1} g(z_j) w_{j,i}^{(2)} \right)$$

# Hidden Layers

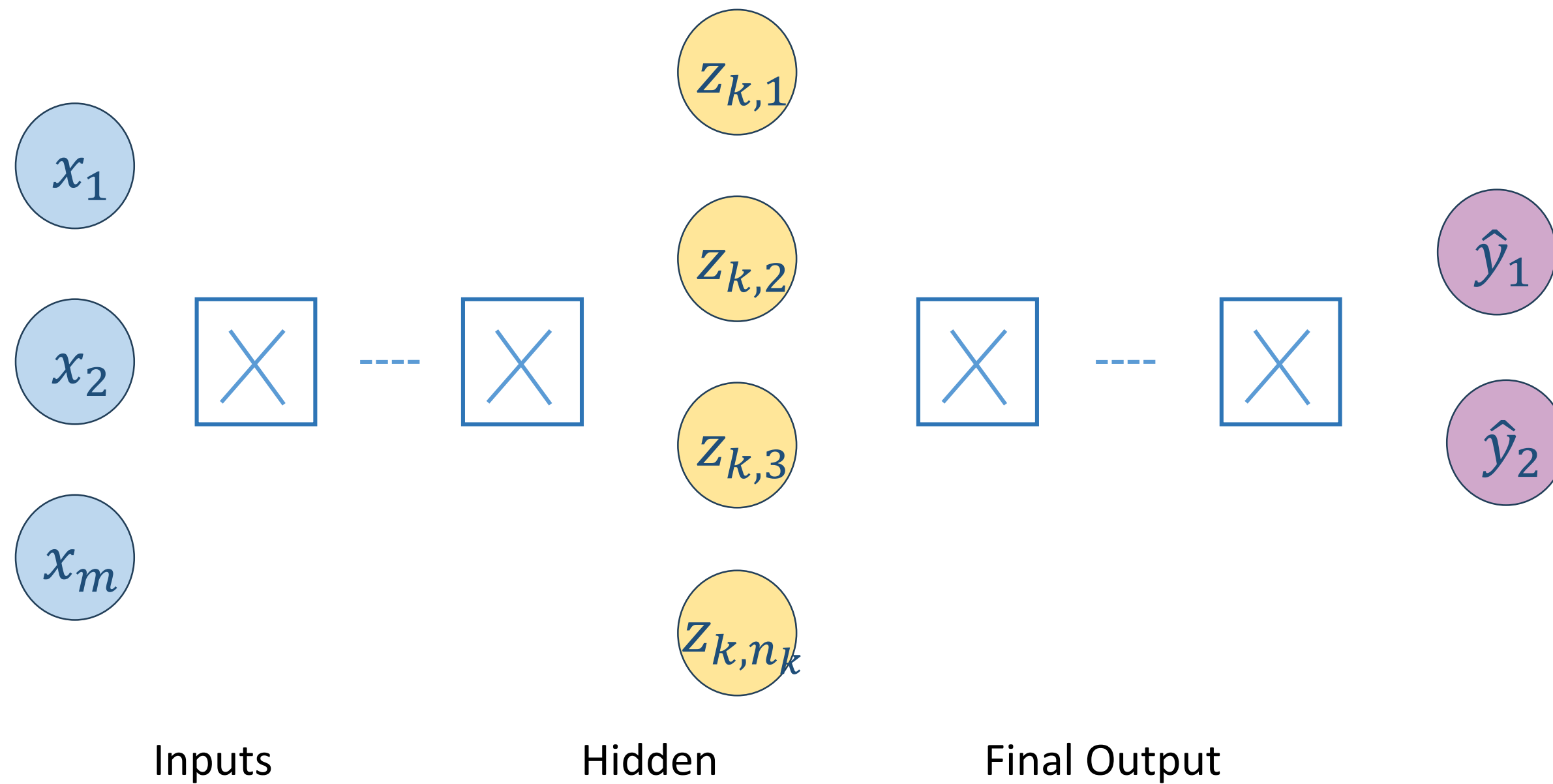


$$z_2 = w_{0,2}^{(1)} + \sum_{j=1}^m x_j w_{j,2}^{(1)} = w_{0,2}^{(1)} + x_1 w_{1,2}^{(1)} + x_2 w_{2,2}^{(1)} + x_m w_{m,2}^{(1)}$$

# Multiple Output Perceptron

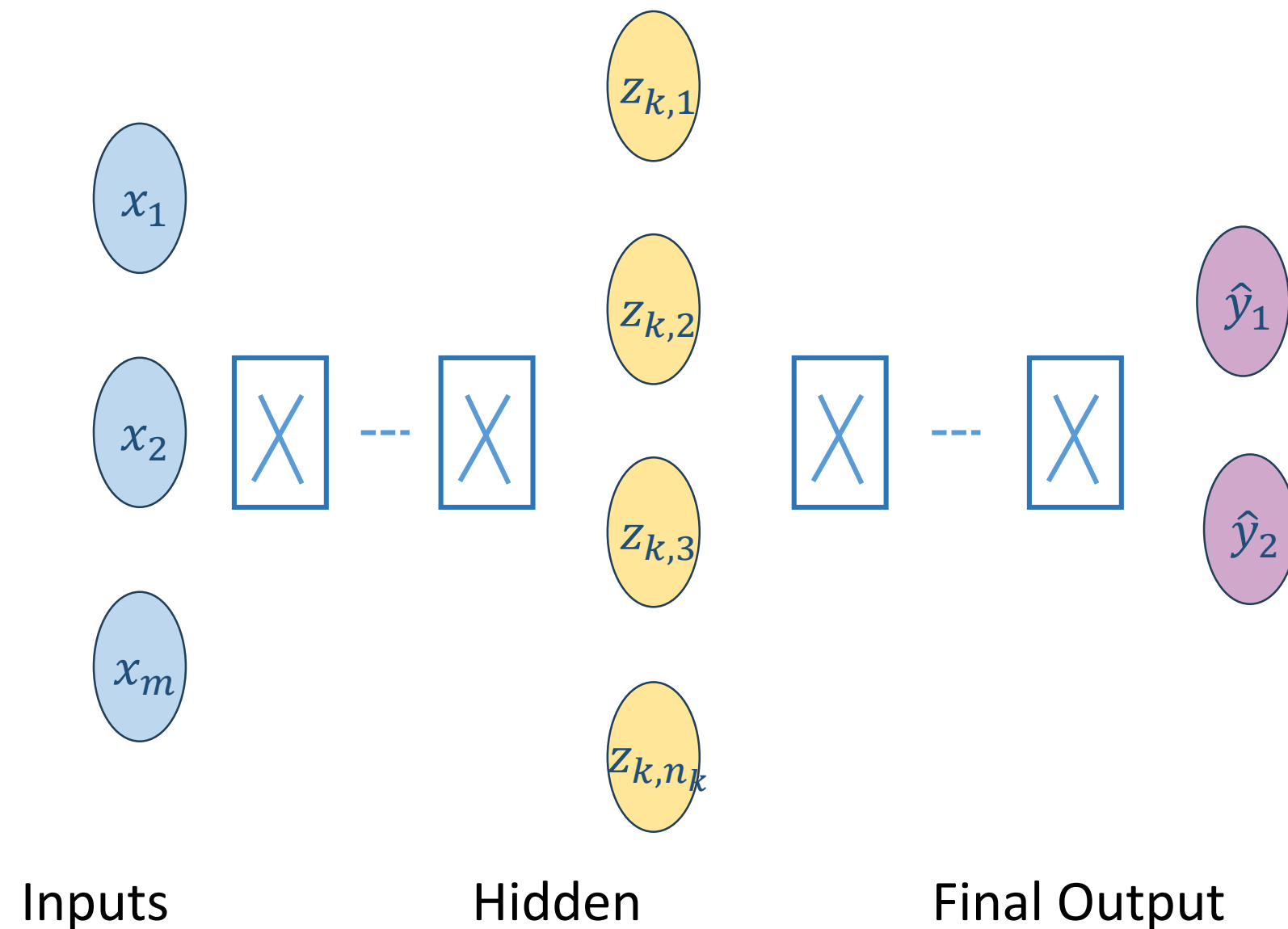


# Deep Neural Network



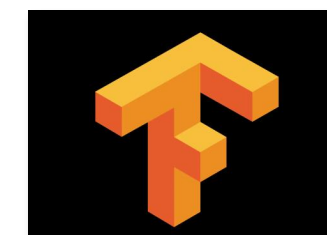
$$z_{k,i} = w_{0,i}^{(k)} + \sum_{j=1}^{n_{k-1}} g(z_{k-1,j}) w_{j,i}^{(k)}$$

# Deep Neural Network



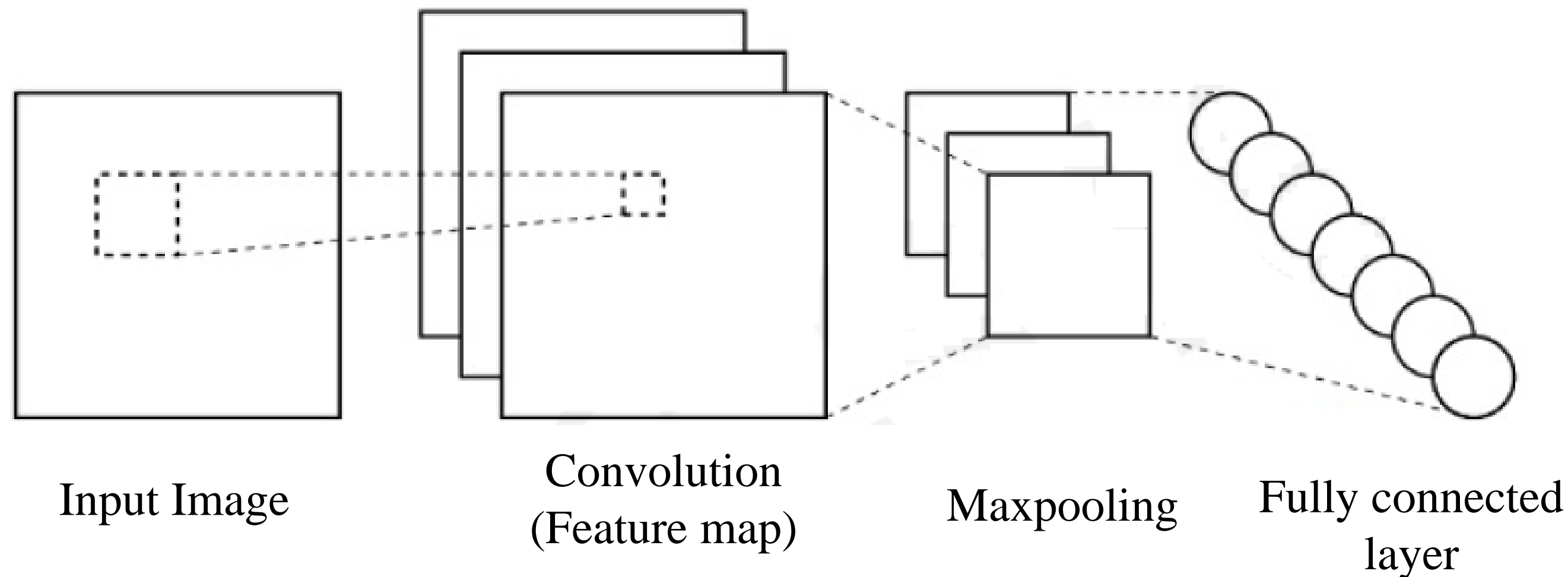
```
import tensorflow as tf

model = tf.keras.Sequential ([
    tf.keras.layers.Dense(n1),
    tf.keras.layers.Dense(n2),
    .
    .
    .
    tf.keras.layers.Dense(2)
])
```



$$z_{k,i} = w_{0,i}^{(k)} + \sum_{j=1}^{n_{k-1}} g(z_{k-1,j}) w_{j,i}^{(k)}$$

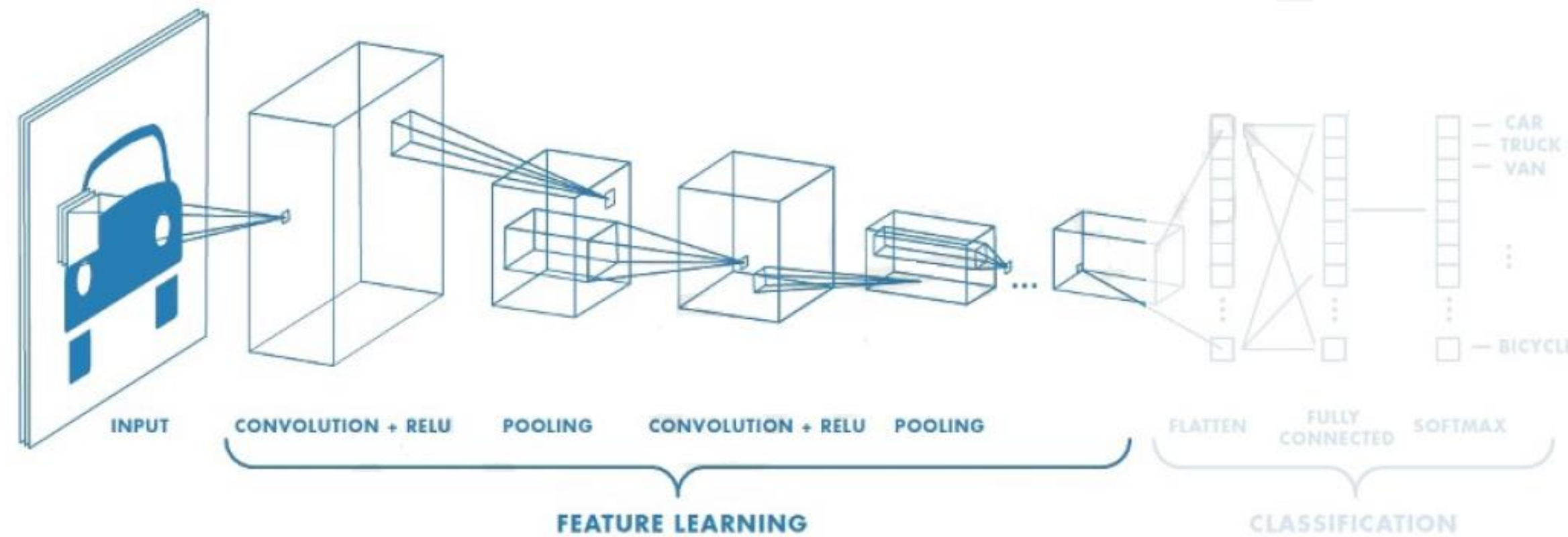
# Convolutional Neural Networks (CNNs)



1. Convolution: Apply filters to generate feature maps
2. Non-linearity: Often ReLU.
3. Pooling: Downsampling operation on each feature map.

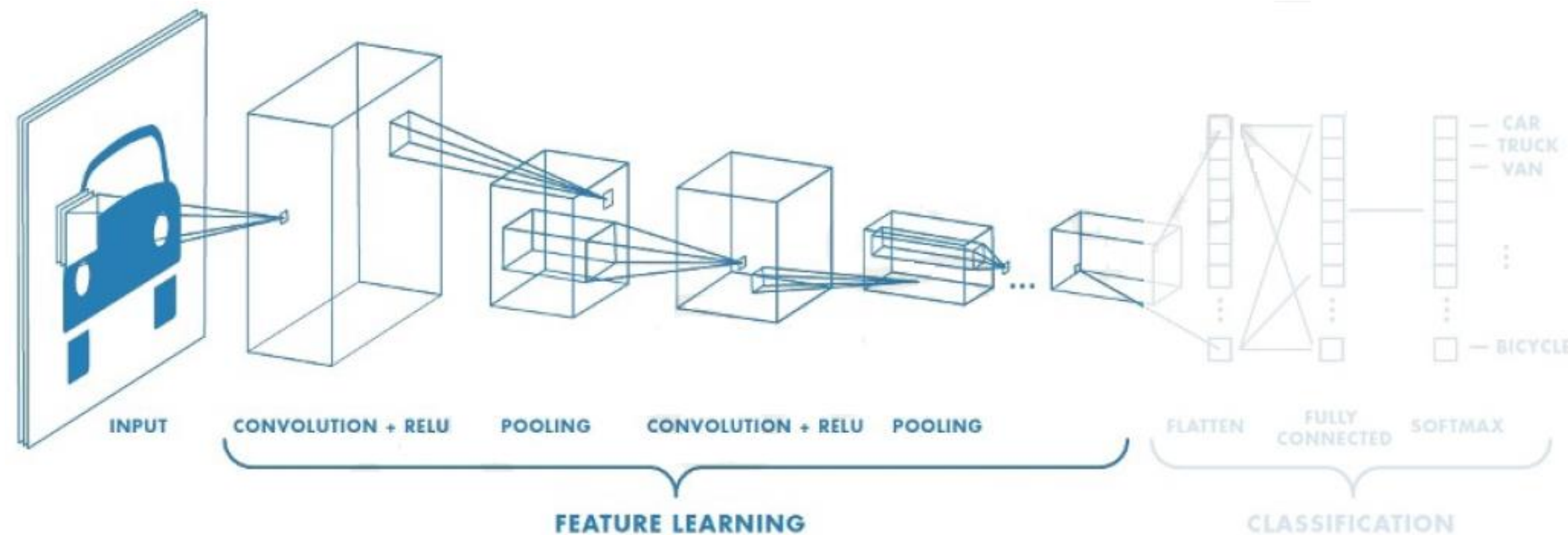
```
tf.keras.layers.Conv2D  
tf.keras.activations.*  
tf.keras.layers.MaxPool2D
```

# CNNs for Classification: Feature Learning



- 1) Learn features in input image through **convolution**
- 2) Introduce **non-linearity** through activation function (real-world data in non-linear!)
- 3) Reduce dimensionality and preserve spatial invariance with **pooling**

# CNNs for Classification: Class Probabilities



- CONV and POOL layers output high-level features of input
- Fully connected layer uses these features for classifying input image
- Express output as probability of image belonging to a particular class

$$\text{softmax}(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

# Classification: Breast Cancer Screening

nature

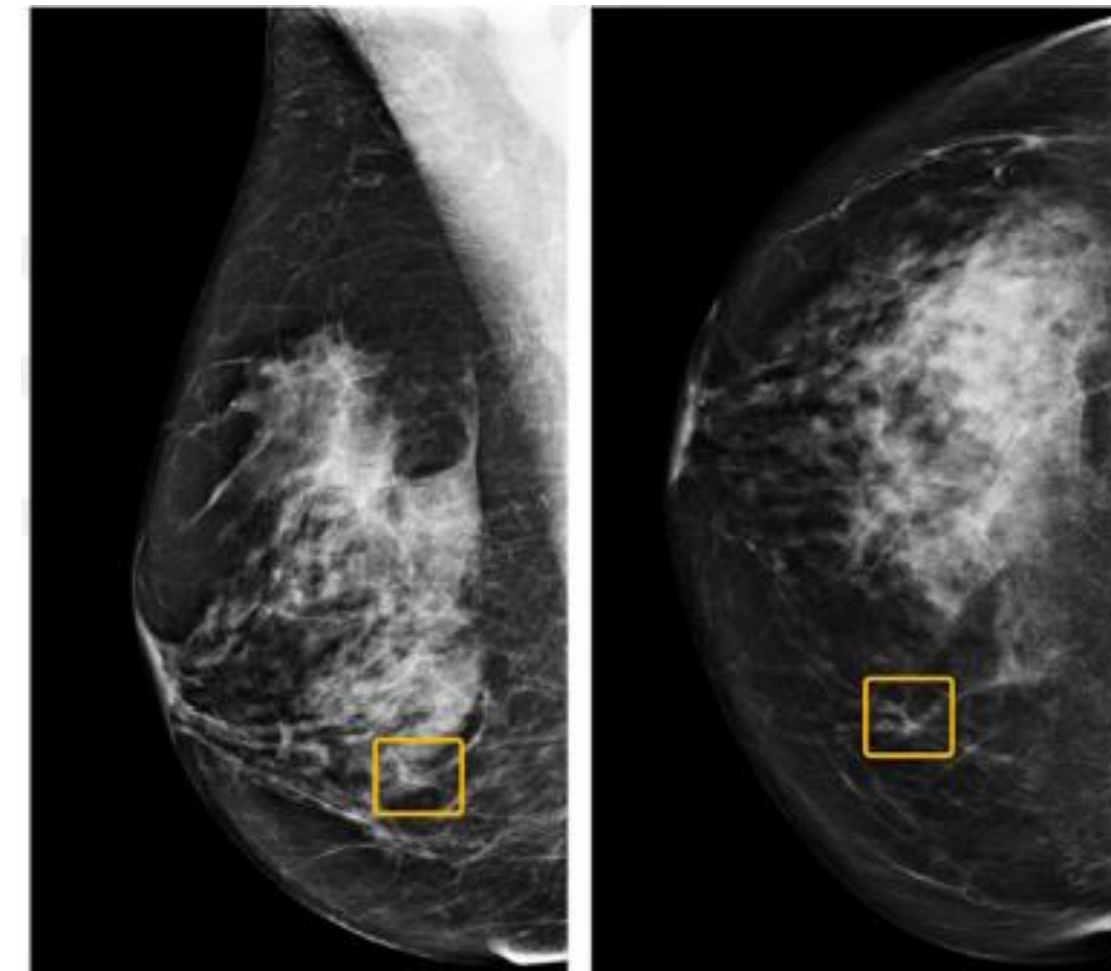
Article | Published: 01 January 2020

## International evaluation of an AI system for breast cancer screening

[Scott Mayer McKinney](#) , [Marcin Sieniek](#), [Varun Godbole](#), [Jonathan Godwin](#), [Natasha Antropova](#), [Hutan Ashrafian](#), [Trevor Back](#), [Mary Chesus](#), [Greg S. Corrado](#), [Ara Darzi](#), [Mozziyar Etemadi](#), [Florencia Garcia-Vicente](#), [Fiona J. Gilbert](#), [Mark Halling-Brown](#), [Demis Hassabis](#), [Sunny Jansen](#), [Alan Karthikesalingam](#), [Christopher J. Kelly](#), [Dominic King](#), [Joseph R. Ledsam](#), [David Melnick](#), [Hormuz Mostofi](#), [Lily Peng](#), [Joshua Jay Reicher](#), ... [Shravya Shetty](#)  [+ Show authors](#)

[Nature](#) 577, 89–94 (2020) | [Cite this article](#)

93k Accesses | 1118 Citations | 3926 Altmetric | [Metrics](#)

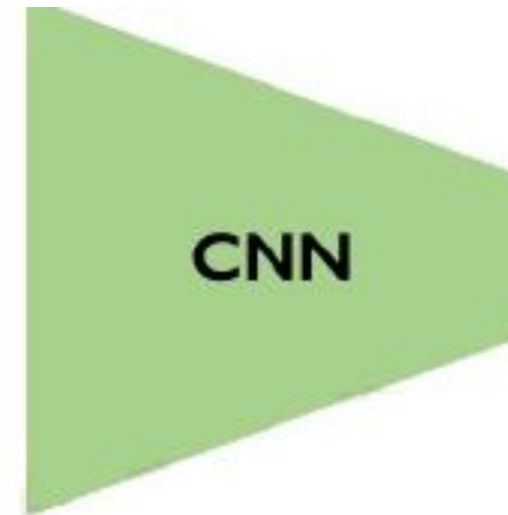


CNN-based system outperformed expert radiologists  
at detecting breast cancer from mammograms

# Object Detection



Image  $x$



CNN

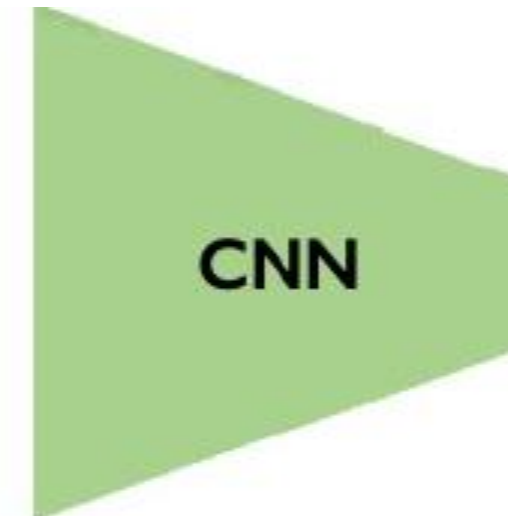


Taxi

Class label  $y$



Image  $x$



CNN

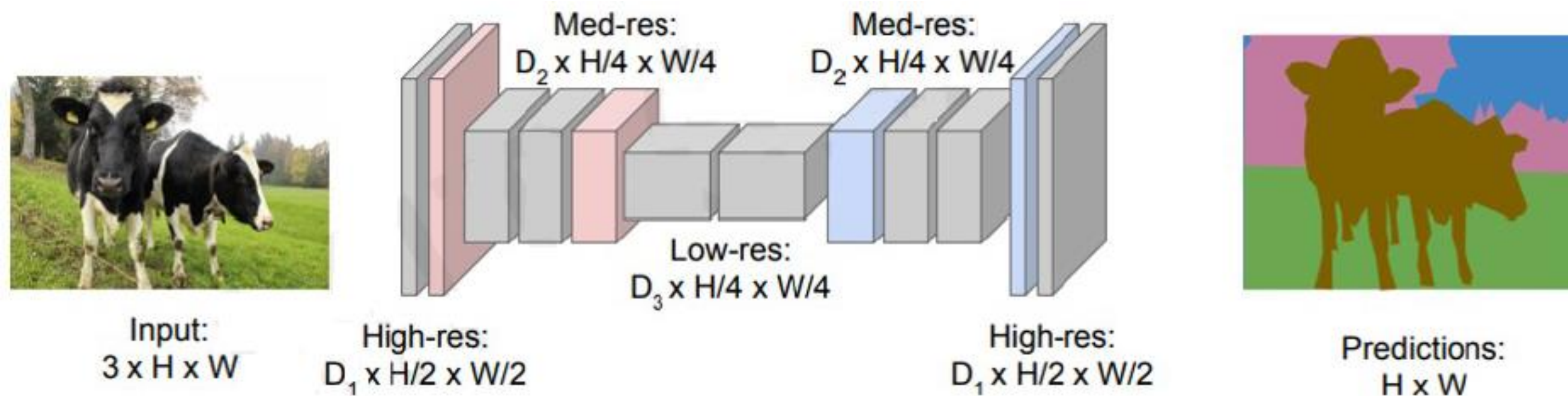


Label  $(x, y, w, h)$

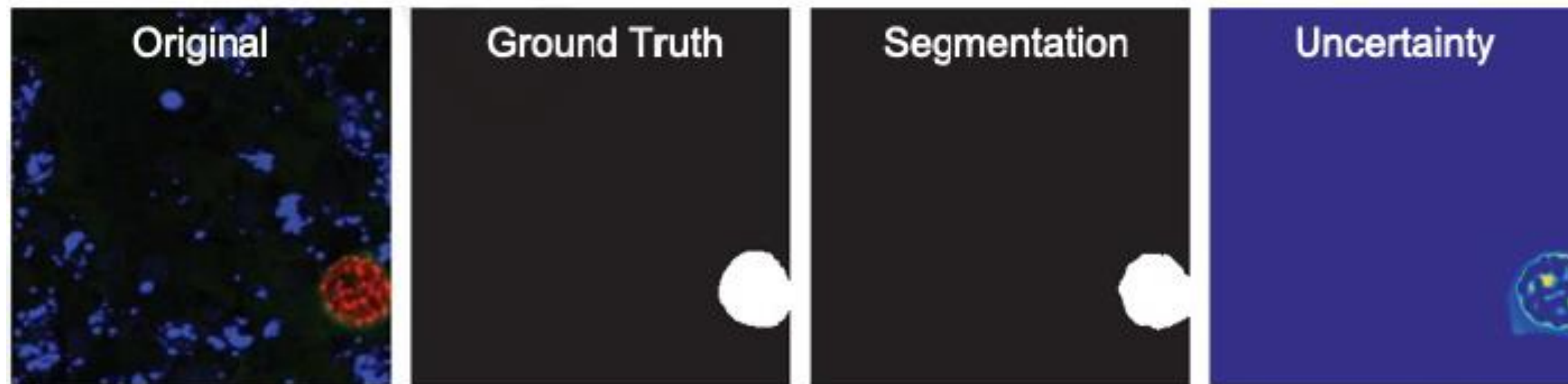
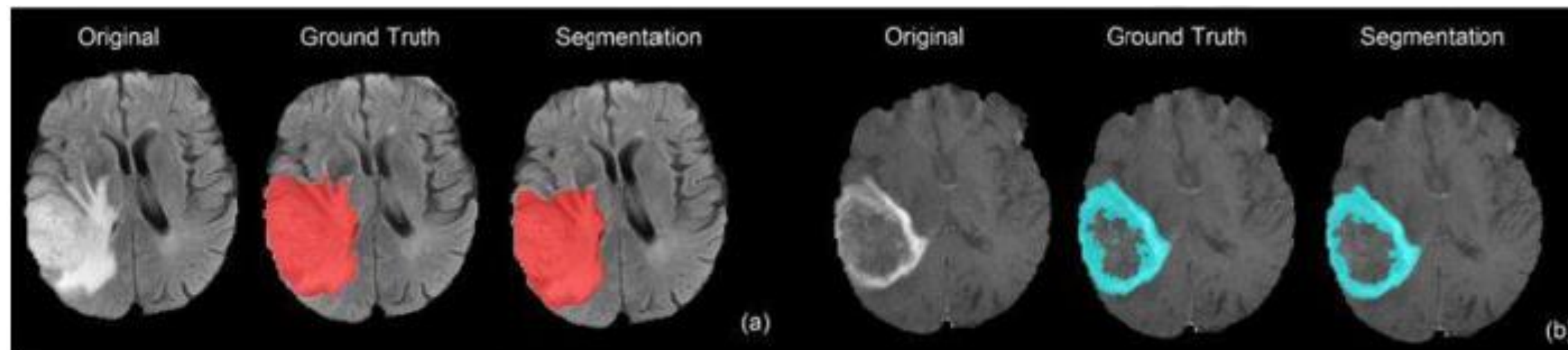
# Semantic Segmentation: Fully Convolutional Networks

FCN: Fully Convolutional Network.

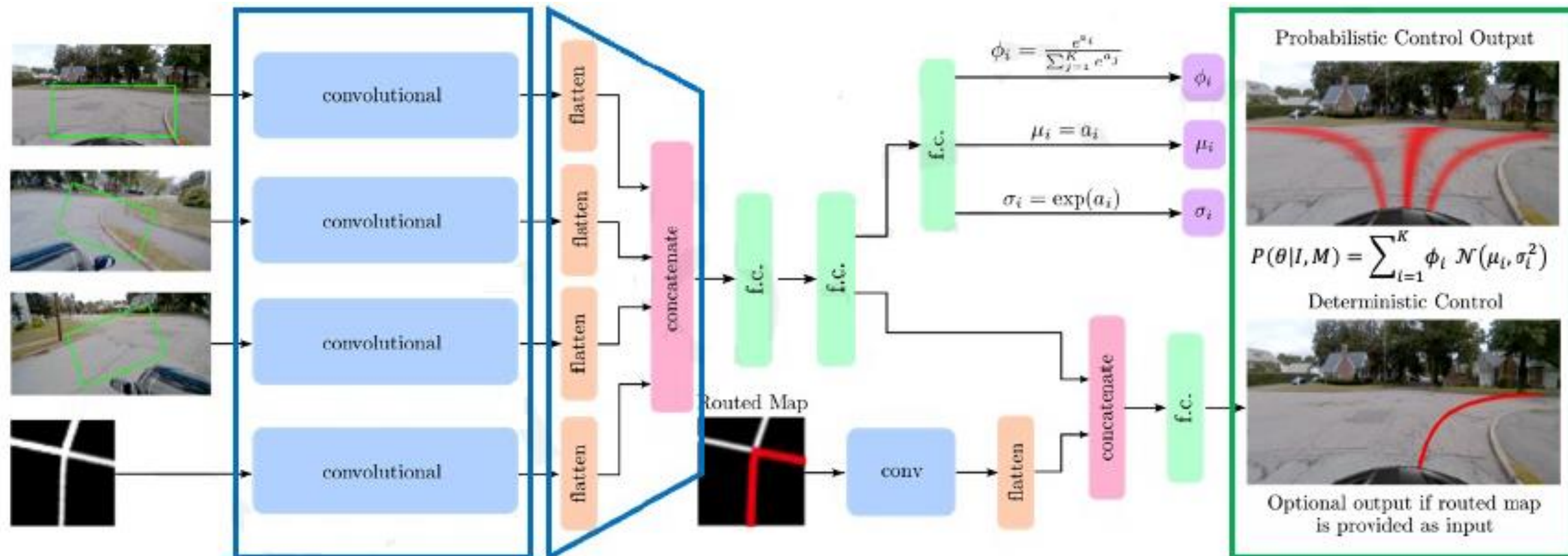
Network designed with all convolutional layers, with downsampling and upsampling operations

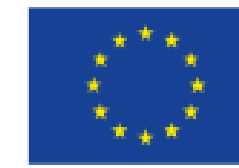


# Semantic Segmentation: Biomedical Image Analysis



# Continuous Control: Navigation from Vision





# References

- Deep Learning, Ian Goodfellow, Yoshua Bengio and Aaron Courville, MIT Press, 2017.
- Fundamentals of Deep Learning (2nd Edition), Nithin Buduma, Nikhil Buduma and Joe Papa, O'Reilly Media, 2022



**FACULTY OF  
ENGINEERING**

**RL4Eng**



Co-funded by the  
Erasmus+ Programme  
of the European Union

**THANK YOU**